Using Profiling Metrics to Categorise Peer Review Types in the Android Project

Raula Gaikovina Kula, Carmago Cruz Ana E., Norihiro Yoshida,
Kazuki Hamasaki, Kenji Fujiwara, Xin Yang and Hajimu Iida
NAIST, JAPAN
{*raula-k, carmago, yoshida, kazuki-h, kenji-f, xin-y*}*@is.naist.jp, iida@itc.naist.jp*

***Abstract*—Profiling of open source software (OSS) peer reviewers have many potential benefits related to quality assurance at both software and project levels. In this paper, we investigate its benefits such as identifying hidden experts, identify inactive or disinterested members to gauge the health of the OSS project and assisting aspiring members to monitor performance, identifying opportunities for career improvement. Preliminary results are promising, proving that our categories are practical and opening many avenues for future work. Recent improvements of data repositories mining tools and techniques make this research timely in able to provide quantitative insights for OSS peer review projects.**

***Keywords*-OSS Peer Review Process; Software Quality Assurance; Knowledge Management; Data Mining**

I. Introduction

In a software project team it is assumed that the most skilled and knowledgeable members come through `experience`, defined by their length of membership *tenureship* and their historical activities. In a code review setting they are known as *experts*, assumed to have high review activity participation. They are efficient, qualified and suitable to inspect, verify and accept any new code changes (referred to as *patches*) merged into source code. They ensure that the quality and reliability of the code is maintained at high standards. However, these experts are usually scarce and unavailable, due to the high workloads.

This effect is even more notorious in the Open Source Software (OSS) project setting. In most instances, members are physically distributed, without face-to-face communication. In recent times, OSS projects have evolved into large and fairly complex systems, arguable competing with their commercial counterparts.

Since the known experts cannot review every patch, the next best option would be finding potential or hidden experts. For example, a potential candidate could be a skilled but relatively new member exhibiting high peer review activities. Identification of such contributors is beneficial, as it would help reduce expert's workloads, improve review times as well as sustain and maintain interests of these potential core members.

Profiling members could also help us to understand the overall health of the OSS project. This is because member's active participation and interest is vital to the livelihood of the project. Bird et al. [1] states that the vitality of a OSS project depends on *"it's ability to attract, absorb and retain developers or face stagnancy and failure"*. Since OSS members are motivated purely by self-interest [2], profiling could help in two ways. Firstly from a management point of view, we could classify inactive members. An increase in this category of members could be an indicator of the degrading health of the project. Secondly, by classifying expert types, we can study their review habits and activities. Profiling could provide insights into how aspiring members advance in their OSS career. This new viewpoint is currently investigated by Capiluppi [3]. Profiling could identify performance indicators, that could be used for career advancement prediction models.

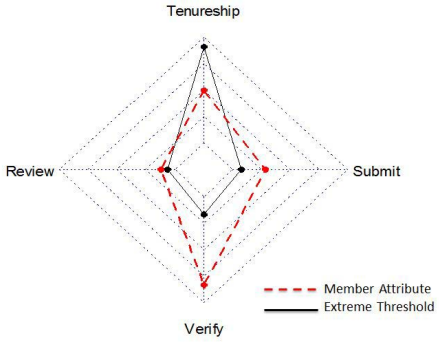

Figure 1. In this example, the profiled contributor has outstanding reviews and verification and submissions only.

In this paper, we aim to profile and categorise OSS members based on their historical activities. We propose a set of key attributes that can be used as profile indicators. In this study, the attributes we measure are based on the members experience. As shown in Fig. 1, the profiling chart illustrates the application of our proposed metrics. The plot with the black lines indicate the threshold levels for each attribute. We measure the following proposed activity metrics (defined in section 2); *tenureship, submit rate, review rate and verify rate*.

To identify outstanding attributes, we introduce project-specific *extreme thresholds*. Extreme thresholds serve to identify outstanding activity levels. For instance in Fig. 1, we know that the member is active (extreme review, verify and submit levels), even-though she has less relatively

978-0-7695-4928-6/12 $26.00 © 2012 IEEE
DOI 10.1109/ISSREW.2012.50

less experience (tenureship). The baselines for the extreme thresholds are based on the 20-80% pareto principle.

To guide our research, we formulated the following research questions:

- **RQ1.** Can we identify and categorise hidden experts?
- **RQ2.** Can we identify inactive or members with declining interest in the project?
- **RQ3.** Does our expert classification provide practical expert classifications?

We studied the Android Open Source Project (AOSP) to evaluate our approach. Our main contributions and findings can be summarized as follows:

1) Profiling and categorised Android members based on contributor activity using our approach
2) Identified two possible member types (senior and senior verifiers) as indicators for the health of an OSS project.
3) The identification of hidden/potential experts. Using our metrics, we were able to identify new members with high activity levels. An example is shown in Fig. 6.

II. Android Open Source Project

The Android Open Source Project (AOSP) is a linux-based operating system for mobile devices such as smartphones and tablet computers, developed by Google in conjunction with the Open Handset Alliance [1]. The first Android-powered smartphones were sold in Q1 2009, and has since grown to become the biggest smartphone operating system.

AOSP currently has a public and private branch for members to contribute patches. In this study, we solely focus on the public branch. Using custom scripts, peer review data was extracted from the online android gerrit code review system [2].

From the AOSP documentation [3], we present the following terms to be used throughout the paper:

- **Member.** An individual who takes on at least one of the following roles: contributor, reviewer, or verifiers/approvers.
- **Peer Review Activities.** An activity performed by peer review members. These activities are classified into the roles of: i). contributor, ii). Reviewers and iii). Verifiers/Approvers. In this study we consider the activity of submitting patches as part of the peer review process, as the promotion of becoming a verifier involves notable contributions of high quality code.
- **Contributors.** A `contributor` is a member that makes contributions to the source code, thus submitters of patches. In this paper, we assume that the contributor is the owner of the code, which in reality is not always the case.
- **Reviewers.** `Code reviewers` are any member / contributor of the project that wishes to review the code. Reviewers usually comment on the code.
- **Verifiers and Approvers.** `Verifiers` are responsible for testing to validate the patches. Contributors can be invited to become verifiers after contributions of significant high-quality code. `Approvers` are experienced members who are said to have made significant contributions and were previously verifiers. Since this is an invite only role and has the power to include of exclude changes, verifiers as assumed to have a higher social status over members.

AOSP uses the GIT source code management system in conjunction with gerrit, a web-based collaborative code review tool. Gerrit automatically records and tracks all merges into the source code, including details related to the code patch and the peer review process activities. To track all peer review activities, we extracted all patches regardless of current state (abandoned, merged or still open). For each patch report, we used specific features for our analysis. For identification, we extracted *the patch id* as well as the *member name, id* and the *email address* of the patch owner. We then extracted all the contributors involved in that patch report, identified as *submitters, reviewers or verifiers.*

We used relational databases to analyse our data and the R Tool [4] for statistical analysis and to generate our profiling charts [5]. Our compiled dataset is readily available online for download[6].

III. Peer Review Profiling

Profiling Metrics

According to Mockus [4], expertise is strongly related with experience. Building on this, we based all of our metrics on membership duration. Also, Bird [5] states that member's review activities follow a hazard rate, dropping activity levels after attainment of high social status. To incorporate this phenomena, our proposed metrics are designed to provide a contributor's activity performance in relation to their tenureship. We measure the activities as a daily rate, thus identifying inactively of members over time. We defined the term *outstanding* as having attributes that exceed (outliers) the central tendency measures.

Each metric is defined using the following attributes:

- **Tenureship** - This is a measure of duration (per day) since a member had joined the project, from the first day till the latest update. Three different review activities are considered: i). review of a patch, ii). submitting

[1] http://source.android.com/
[2] https://android-review.googlesource.com/
[3] http://source.android.com/source/roles.html
[4] http://www.r-project.org/
[5] http://cran.r-project.org/web/packages/fmsb/fmsb.pdf
[6] http://sdlab.naist.jp/reviewmining/

Table I
EXPERT MATRIX: T= TENURESHIP, S=SUBMISSIONS, R=REVIEWS, V=VERIFICATIONS, X= EXTREME ATTRIBUTE

Expert	Member Types	T	S	R	V
Verifier	**Core Member (CM)**	X	X	X	X
Verifier	**SeniorVerifyingSubmitter(TVS)**	X	X		X
Verifier	**SeniorVerifyingReviewer(TVR)**	X		X	X
Verifier	**ActiveMember(VSR)**		X	X	X
Non-verifier	SeniorSubmittingReviewer(TSR)	X	X	X	
Verifier	**SeniorVerifier(TV)**	X			X
Verifier	**VerifyingSubmitter(VS)**		X		X
Verifier	**VerifyingReviewer(VR)**			X	X
Non-verifier	SeniorSubmitter(TS)		X	X	
Non-verifier	SubmittingReviewer(SR)		X	X	
Non-verifier	SeniorReviewer(TR)	X		X	
Verifier	**Verifier(V)**				X
Non-verifier	Reviewer(R)			X	
Non-verifier	Senior(T)	X			
Non-verifier	Submitter(S)		X		

a patch, or iii). verification of a patch. Members with outstanding tenureship are referred to as *Seniors (T)*.

- **Submit rate** - This is a measure of how many patches the member has submitted. We called members with outstanding submissions *Submitter*.
  The derived metric is defined as:
$$\frac{numberofSubmissions}{Tenureship} \quad (1)$$
- **Review rate** - This is a measure of how many patches the member has been involved as a reviewer. We call members with outstanding reviews *Reviewer*.
  The derived metric is defined as:
$$\frac{numberofReviews}{Tenureship} \quad (2)$$
- **Verify rate** - This is a measure of how many patches a contributor has verified and approved before merging into the source code. We refer to these members as *Verifier*. We assume that experts should be at least verifiers, as they are able to approve new code. The derived metric is defined as:
$$\frac{numberofVerifications}{Tenureship} \quad (3)$$

Table I shows a matrix of all the possible combinations of the proposed metrics, from which we derived different *member types*. From our metrics, were identified a combination of 15 different member types, each categorised by the number of extreme attributes identified.

Table I also shows that 8 out of the 15 classifications are verifiers. As mentioned in the previous section, verifiers have higher authority, thus could be referred to as being the experts.

Both member types senior and senior verifiers could possibly contain inactive or members with declining interest in the project. This is because for long tenured members, it is rather usual not to exhibit other extreme activities.

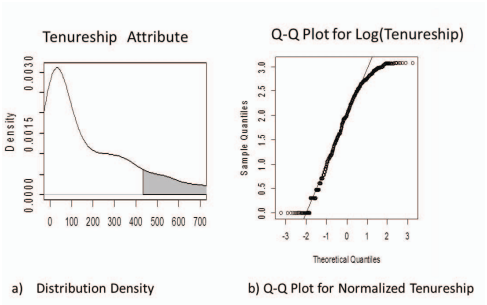

Figure 2. Each graph represents the distribution of the tenureship attributes. The shape of the density suggest a positively skewed normal distribution. The shaded area highlights the experts for AOSP.

Extreme Thresholds

In order to identify members with specific outstanding activity attributes, we define *extreme thresholds* applying the `pareto principle`, as our baseline measure. The pareto rule, a concept from economics with applications in engineering, states that roughly 80% of the effects come from 20% of the causes.

Activity rates can be misleading for new members, due to their low tenureship. We defined a minimal threshold so that a member is only eligible for categorisation after a defined tenureship duration. We applied the same 20-80% ratio. Thus, members are required to have tenureship greater than 20% of the population before being eligible.

IV. PRELIMINARY RESULTS

Using our custom scripts, we extracted 11,512 patch reports over a 38 month period (2008/10/21 - 2012/01/27). During this time, AOSP recorded 1,040 members.

Extreme threshold evaluation

The pareto principle can be expressed mathematically as a power-law or log-normal distribution. Log normal distributions are characterized for having only positive values and are skewed with long tails. Moreover, they must be log-normally distributed [7]. As an example, we show in Fig. 2 (a) the skewed distribution of one of our attributes, tenureship. To prove our attributes were log normal distributed, we use Q-Q probability plots, as shown in Fig. 2(b). In Fig. 2 (a), the shaded area represents the seniors (extreme thresholds of tenureship) of AOSP.

We verified the pareto principle for all metrics before calculating the extreme thresholds. Results are shown in Table II. It is interesting that the verify threshold is 0%, this validates that verfiers are automatically extreme members of the project. In Table II, it is shown that members must have at least 12 days of tenureship before considered eligible for classification.

[7]a random variable X is log-normally distributed if log(X) is normally distributed

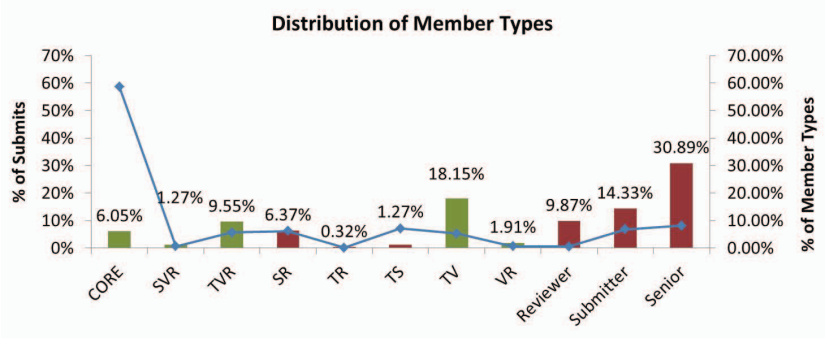

Figure 5. This figure shows the distribution of expert types. Expert types with no representation (0%) are in the light blue box.

Table II
EXPERT THRESHOLDS FOR AOSP

Tenureship (Days)	Submit Rate (DailyRate)	Review Rate (DailyRate)	Verify Rate (DailyRate)	Eligibility (Days)
430.5	0.100	0.121	0	12

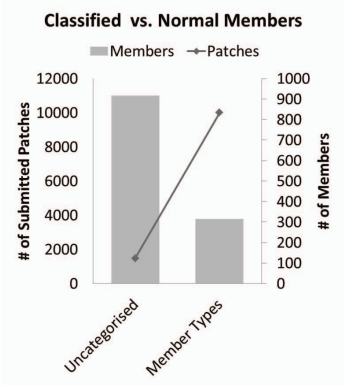

Figure 3. The figure illustrates comparison between the uncategorised and the member types of AOSP.

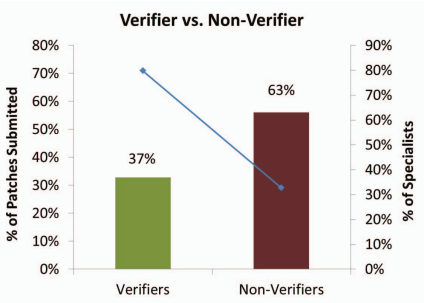

Figure 4. The figure illustrates that in AOSP, there are more non-verifiers than verifiers, however more patches are submitted by verifiers.

Table III
PEARSON COR. MATRIX: T= TENURESHIP, S=SUBMIT, R=REVIEW, V=VERIFY,(VERIFIER'S COR. IN BRACKETS)

	S	R	V
S	-	0.063(0.36)	0.026(0.32)
R	0.063(0.36)	-	0.31(0.90)
V	0.026(0.32)	0.31(0.90)	-

Member type analysis

As seen in Fig. 3, there are considerably more uncategorised members (917 members) in comparison to member types (314 members). However in contrast, a total of 10,050 patch reports were submitted by the member types as opposed to 1,582 patch reports submitted by non-categorised members. This illustrates that experts, even though a minority, contribute the most to the project. This is consistent of the OSS onion model [6], in which with the minority of inner layers taking more leading and contributing roles than the outer layers.

In Fig. 4, we grouped together the expert types (with verifier related types as shown in Table I) against the non-verifier categories. Consistent with the onion model, verifiers have more patch contributions.

In Fig. 5, we take a closer look at the expert types introduced in this paper (Table I). Out of the 8 expert types, four are not represented (expert verifier, senior verifier, senior verifying submitter and verifying submitter). This suggests that maybe the verification attribute could be strongly correlated with the other metrics. Table III shows the results of the pearson correlation tests between the three attributes. Between member types (shown in brackets), there is very strong correlation (0.9) between verify rates and review rates. This indicates that a verifier's takes part in as much reviews as verifications. The relationships between review activities is still not fully understood and could be another interesting avenue for future work.

As seen in Fig. 5, the two top member types possess the tenureship attribute (seniors with 30% and senior verifiers with 18.2%). From these results we deduce that senior

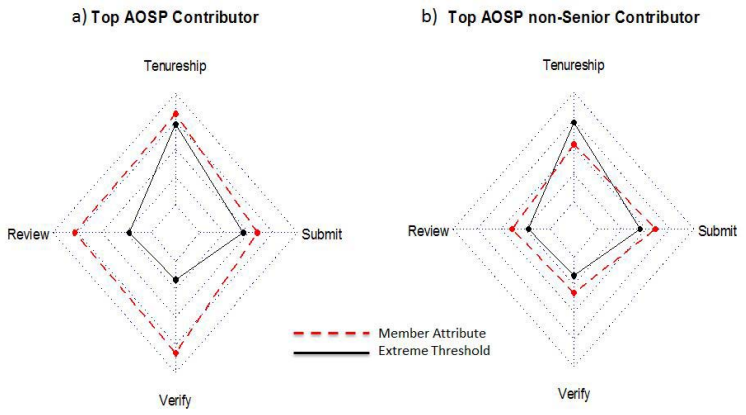

Figure 6. Example showing the performances of a) Top AOSP Contributor and b) A hidden/potential expert (extreme activity although not a senior contributor)

members could be associated with the project maturity. It would be interesting to apply our approach to a less mature project. As for an indication of the current health of the OSS project, we would need to progressively monitor the progressive growth of senior types over an extended period of time. This will be considered for future work.

Finally, core members (with all four extreme attributes) who are considered the most skilled experts with the highest patch submissions, only account for 6.1% of the member types. From a newbie viewpoint, the ultimate career goal would be to reach this type.

AOSP potential experts

One of the main goals of our research was to identify hidden or potential experts. We approached this using the verifier types. Fig. 6 illustrates the identification of the hidden experts. In Fig. 6(a), the profile of contributor with the highest attribute scores is shown. It can be observed that the levels of reviews and verifications are particularly high. On the other hand, Fig. 6(b) represents the profile of a possible potential hidden expert. This is because this contributor also has extreme activity levels, however, is not considered a senior according to our thresholds. We find that this contributor has only been a contributor for 43 days, but has been very active, verifying 42 patch reports, submitting 21 patches and reviewing an additional 41 patch reports. This user could possibly be a newly employed google developer hired for the Android project, however this claim is not validated. Validation of our results is to be approached in future work.

V. Discussion

We view our results as the initial steps towards profiling and categorising different member types. Moreover, we also were interested to identify hidden or potential experts. Our initial results indicate that this can be possibly achieved, however validation of our results with the actual project members is required. Thus, *in response to RQ1, results suggest that our member types are able to identify these experts.*

In Fig. 5, seniors and senior verifiers compose of most (30%) member types. We suspect that both member types could possibly include inactive or decline interest members. Results are inconclusive, however, monitoring both these members types could provide useful towards gauging OSS health. Therefore, *in response to RQ2, although we have not validated the categorisation of inactive members, we believe our metrics could be useful OSS health indicators.*

Our metrics generated 8 possible expert types, however, as seen in Fig. 5 only four expert types have no representation. Results suggest a strong correlation between reviews and verification. Both seniors and senior verifiers types could provide insights into OSS health.

According to the android documentation, verifiers are "invited members that have submitted significant amount of high-quality code to the project and demonstrated their design skills and have made significant technical contributions to the project ". Therefore, comparing and understanding their patch review process and code properties could help aspiring members towards the ultimate goal of being a core member type. Individual profiling also helps a young member track and assess her current performance.

Taking all these points into account, *in response to RQ3, results suggest that our metrics and the proposed member types do have practical applications, however, more work is needed to fully understand all of the derived types.*

VI. Threats to Validity

Currently our thresholds are based on the 20-80% pareto rule. This is only used as an example of a set baseline, however in its defence, as seen in Fig. 3, Fig. 4 and 5 the member types, verifiers and the ratio of the number of core members to their submitted patches are consistently show that most of the activities are performed by a minority. We

believe that the pareto rules could be substituted for other techniques such as Tukey's outlier equation [7].

We identified that the accuracy and validation of our results with the real world is a threat. Specifically the issue of *email aliasing*. By using semi-manual processes of cross-checking the username, name and email address for duplicates, we are confident of our contributors list. As future work, we would like to look at the histories of the android members, enabling validation of our results.

Our next step is to apply our approach with other similar projects to generalise our results. We believe that with more projects, we should be able to redefine our thresholds and better understand our expert types.

VII. Related Work

Much research has been performed on different aspects of OSS. Researchers have focused the OSS review process and organisational structure [8], [9], and its social technical aspects [10], [11]. However, we assume that most studies have not attempted to profile members.

Finding and profiling experts is not a new concept and used in knowledge management and collaboration [12] in other fields such as artificial intelligence. For example, MITRE created an expert finder [8] to help users quickly locate people who know about a particular subject area. Other implementations are with the peer review process for journals and conferences. The Expertise Recommender [13] and Expertise browser [14] are example of approaches to identify implementation experts. Similar to these approaches, we used data mined from the software repositories to determine our member types.

VIII. Conclusion and Open Issues

Currently, we only focus on reviewer profiling, assessing current performance and thier standing in the project. Our overall goal is towards development of an OSS expert recommendation system. We intend to combine historical analysis to find the experts for certain locations of components of the system, knowledgeable areas of the system. Also, we want to track inactive or declining interest members so we can devise counter strategies. Finally, we would like to identify potential "rising stars", providing this career advancement pathways modelled on current experts.

There are still outstanding issues for future work, including the following:

- validation of results with more real world case studies.
- further investigation of the senior and senior verifiers types at different stages of an OSS project.
- investigation of peer review processes and code patches between expert (verifier) and non-expert types.
- improvement of the expert thresholds by exploring validation.

[8]http://www.mitre.org/news/the_ edge/june_98/third.html

References

[1] C. Bird, A. Gourley, and P. Devanbu, "Detecting patch submission and acceptance in oss projects," in *In Proc. MSR'07*, Washington, DC, USA, 2007, pp. 26–29.

[2] K. Lakhani and R. Wolf, "Why hackers do what they do: Understanding motivation and effort in free/open source software projects," in *Perspectives on Free and Open Source Software*. Cambridge, Mass: MIT Press, 2005.

[3] A. Capiluppi, P. Lago, and M. Morisio, "Characteristics of open source projects," in *Proc. CSMR'03*, march 2003, pp. 317 – 327.

[4] A. Mockus and J. D. Herbsleb, "Expertise browser: a quantitative approach to identifying expertise," in *Proc. ICSE '02*. ACM, 2002, pp. 503–512.

[5] C. Bird, A. Gourley, P. Devanbu, A. Swaminathan, and G. Hsu, "Open borders? immigration in open source projects," in *Proc. MSR '07*. Washington, DC, USA: IEEE Computer Society, 2007, pp. 6–14.

[6] K. Crowston and J. Howison, "The social structure of free and open source software development," *First Monday*, vol. 10, no. 2, 2005.

[7] D. C. Hoaglin, F. Mosteller, and J. W. Tukey, "Understanding robust and exploratory data anlysis," *Wiley Series in Probability and Mathematical Statistics, New York: Wiley*, 1983.

[8] J. Asundi and R. Jayant, "Patch review processes in open source software development communities: A comparative case study," in *Proc. HICSS '07*. Washington, DC, USA: IEEE Computer Society, 2007, pp. 166c–.

[9] P. C. Rigby, D. M. German, and M.-A. Storey, "Open source software peer review practices: a case study of the apache server," in *Proc. ICSE'08*, New York, NY, USA, 2008, pp. 541–550.

[10] M. Nurolahzade, S. M. Nasehi, S. H. Khandkar, and S. Rawal, "The role of patch review in software evolution: an analysis of the mozilla firefox," in *Proc. IWPSE-Evol '09*. New York, NY, USA: ACM, 2009, pp. 9–18.

[11] G. von Krogh, S. Spaeth, and K. R. Lakhani, "Community, joining, and specialization in open source software innovation: a case study," *Research Policy*, vol. 32, no. 7, pp. 1217–1241, 2003.

[12] K. Balog and M. de Rijke, "Determining expert profiles (with an application to expert finding)," in *IJCAI*, 2007, pp. 2657–2662.

[13] D. W. McDonald and M. S. Ackerman, "Expertise recommender: a flexible recommendation system and architecture," in *Proceedings of the 2000 ACM conference on Computer supported cooperative work*, ser. CSCW '00. New York, NY, USA: ACM, 2000, pp. 231–240.

[14] A. Mockus, R. T. Fielding, and J. D. Herbsleb, "Two case studies of open source software development: Apache and mozilla," *ACM Trans. Softw. Eng. Methodol.*, vol. 11, no. 3, pp. 309–346, Jul. 2002.